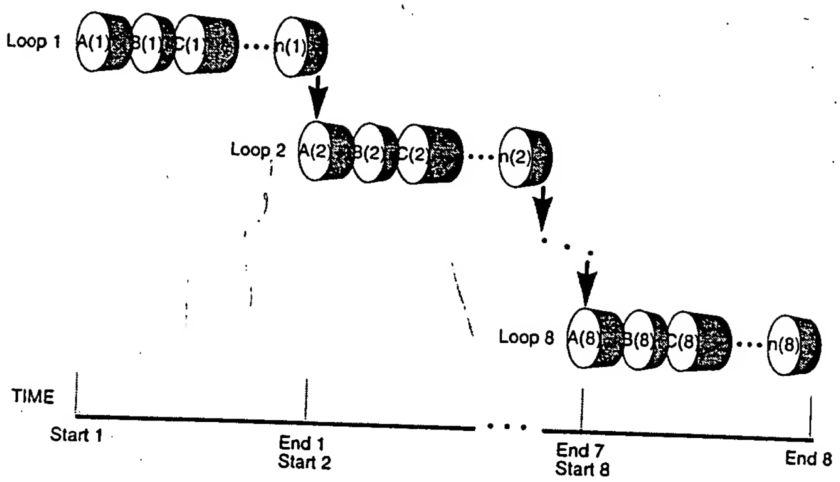

Loop 1
A(1)
B(1)
C(1)
n(1)
Loop 2
A(2)
B(2)
C(2)
n(2)
Loop 8
A(8)
B(8)
C(8)
n(8)
TIME
Start 1
End 1
Start 2
End 7
Start 8
End 8

Figure 1

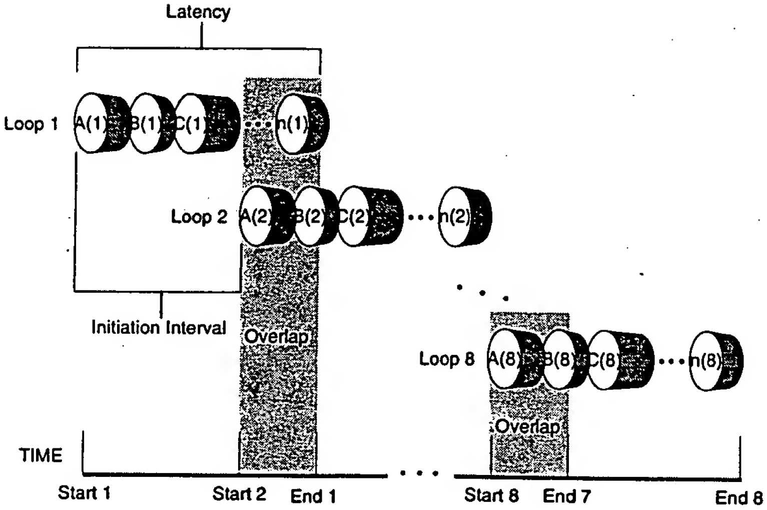

Latency
Loop 1
A(1)
B(1)
C(1)
n(1)
Loop 2
A(2)
B(2)
C(2)
n(2)
Initiation Interval
Overlap
Loop 8
A(8)
B(8)
C(8)
n(8)
Overlap
TIME
Start 1
Start 2
End 1
Start 8
End 7
End 8

Figure 2

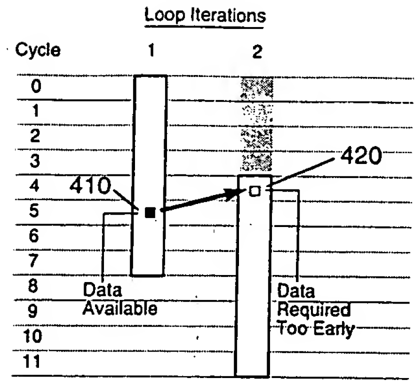


Figure 3 (a)

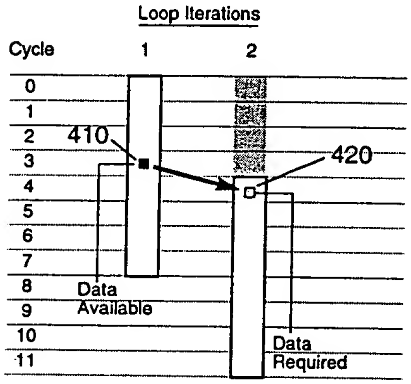
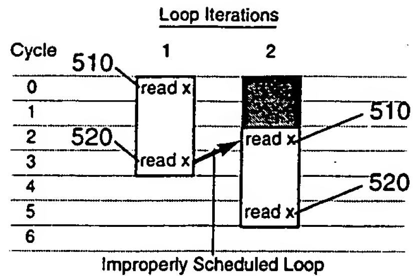
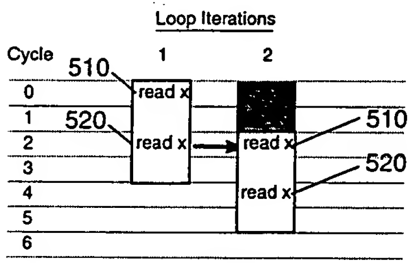


Figure 3 (b)



Improperly Scheduled Loop

Figure 4 (a)



Properly Scheduled Loop

Figure 4 (b)

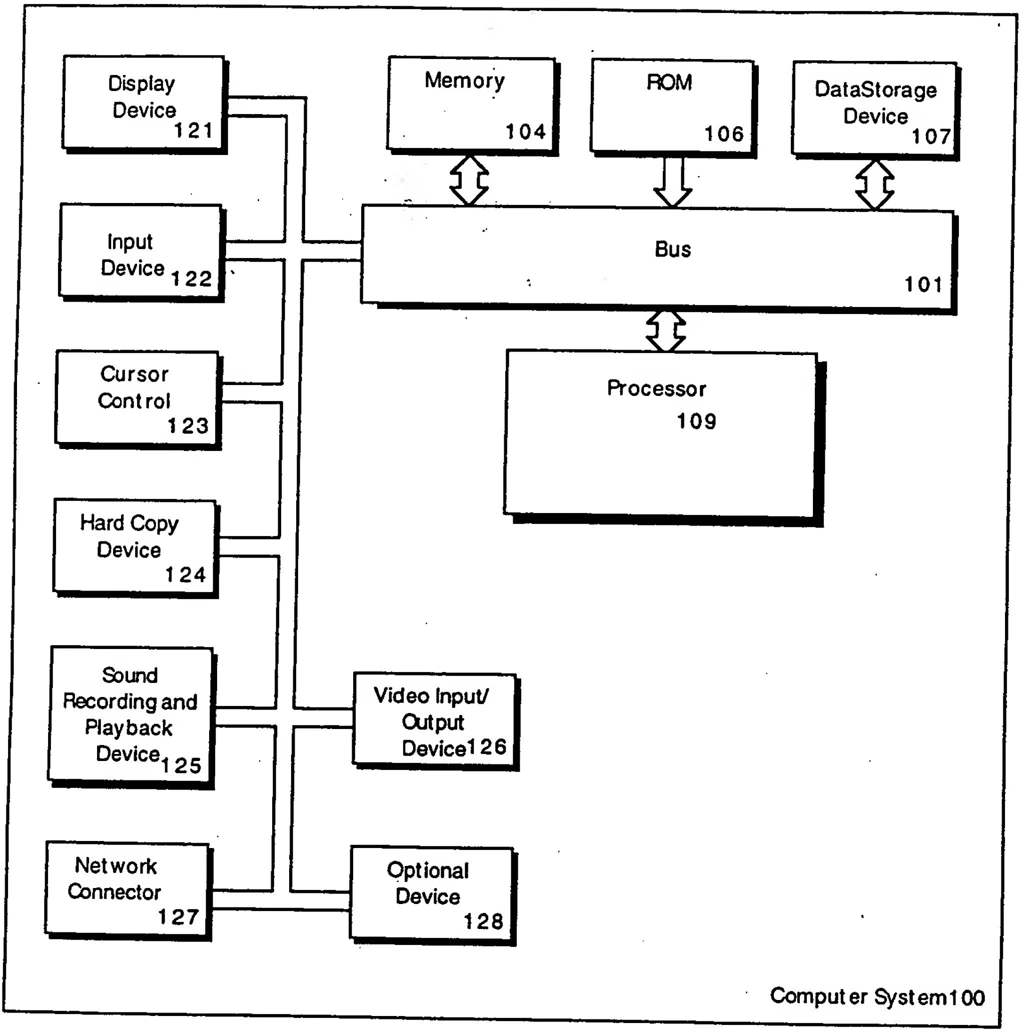

Display Device 121
Memory 104
ROM 106
DataStorage Device 107
Input Device 122
Bus 101
Cursor Control 123
Processor 109
Hard Copy Device 124
Sound Recording and Playback Device 125
Video Input/ Output Device126
Network Connector 127
Optional Device 128
Computer System100

Figure 5

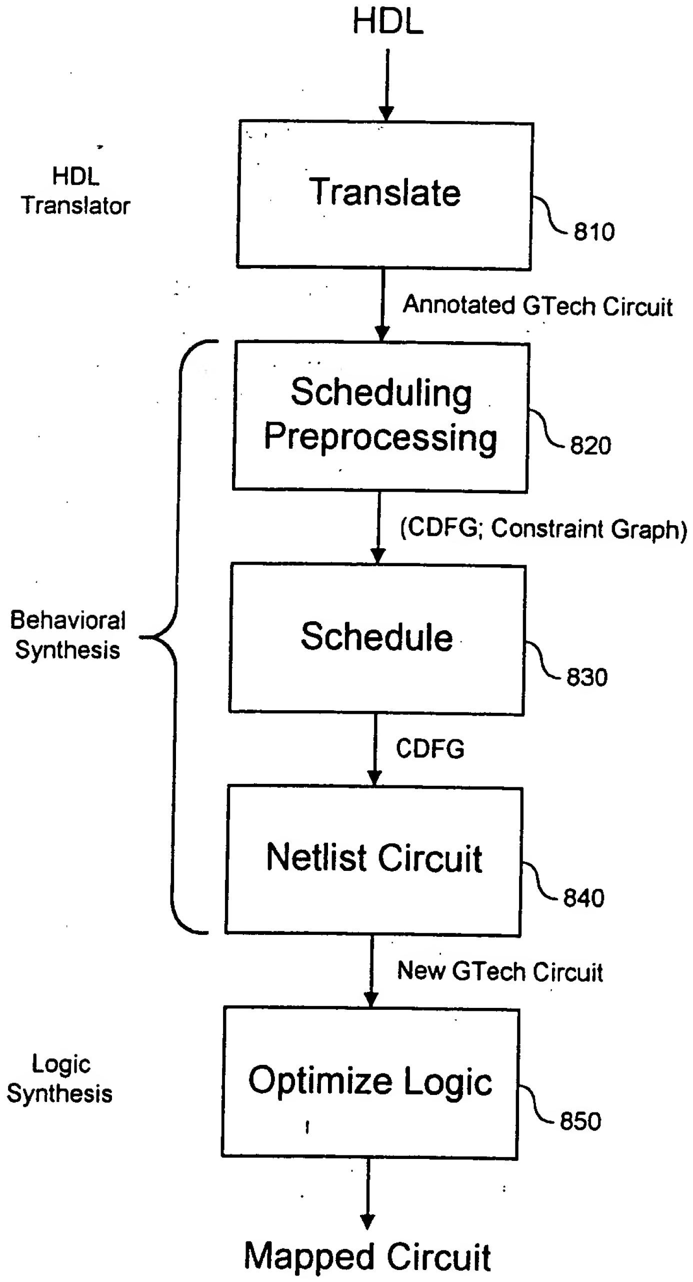


Figure 6 Synthesis with Scheduling

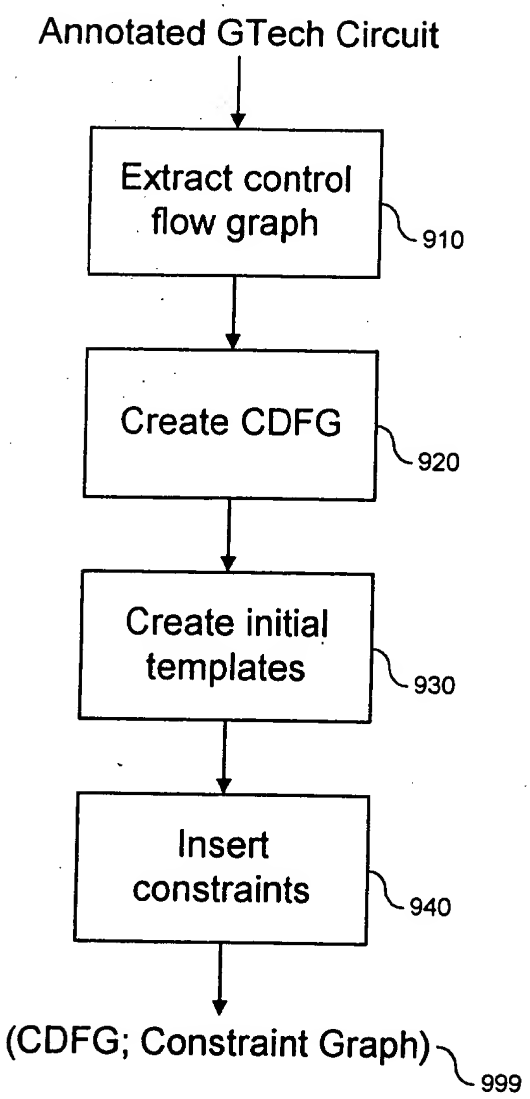


Figure 7

Scheduling Preprocessing

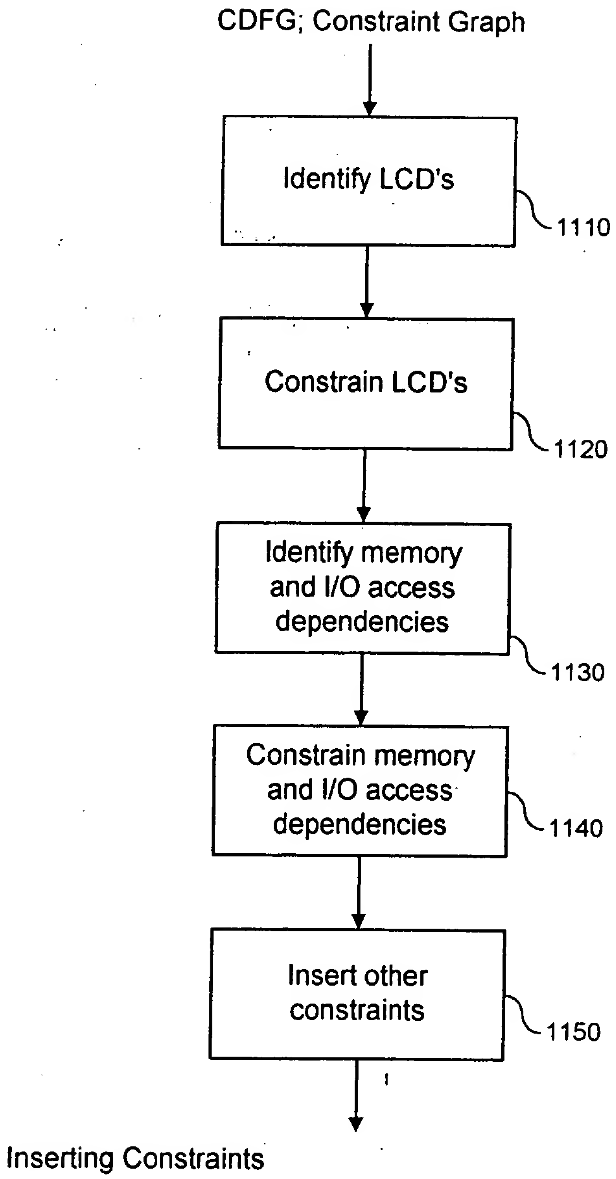
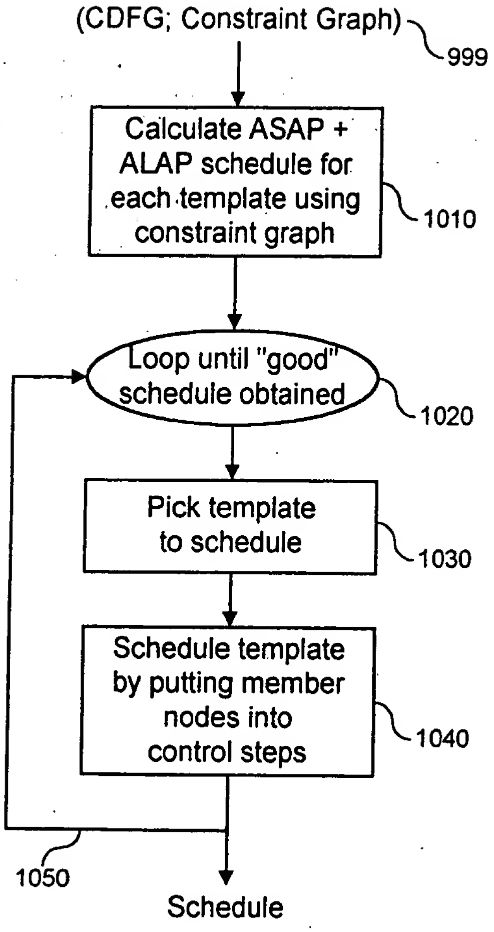

CDFG; Constraint Graph
Identify LCD's
1110
Constrain LCD's
1120
Identify memory and I/O access dependencies
1130
Constrain memory and I/O access dependencies
1140
Insert other constraints
1150
Inserting Constraints

Figure 8


(CDFG; Constraint Graph)
999
Calculate ASAP + ALAP schedule for each template using constraint graph
1010
Loop until "good" schedule obtained
1020
Pick template to schedule
1030
Schedule template by putting member nodes into control steps
1040
1050
Schedule

Scheduling Using Templates

Figure 9

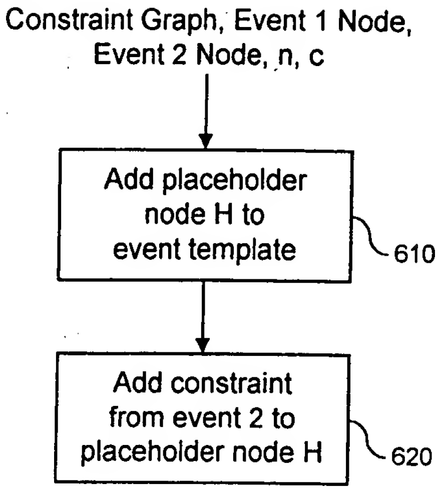

Constraint Graph, Event 1 Node, Event 2 Node, n, c
Add placeholder node H to event template
610
Add constraint from event 2 to placeholder node H
620

Figure 10

```
module loopex8 ( c, x, y, z, clock);
input [1:0] x, y, z;
   input clock ;
   output [2:0] c;
   reg [2:0] c;
   reg [2:0] p;

   always begin                              3030

         forever begin : theloop                       3010

                  c <= x - p ;

                  @(posedge clock) ;                   3020

                  p = y + z ;

                  @(posedge clock) ;

        end

  end

endmodule
```

Figure 11

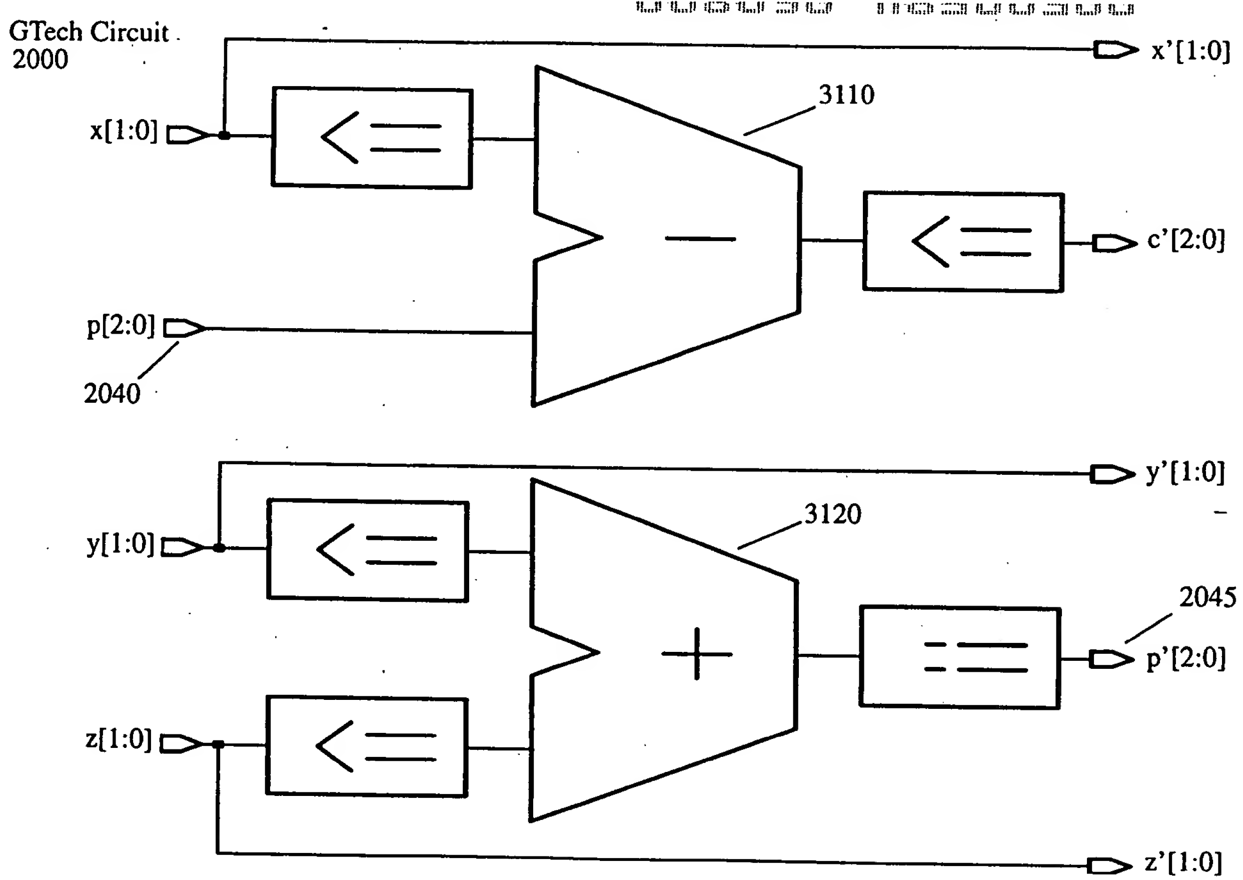

GTech Circuit
2000
x[1:0]
p[2:0]
2040
3110
x'[1:0]
c'[2:0]
y[1:0]
z[1:0]
3120
y'[1:0]
2045
p'[2:0]
z'[1:0]

Figure 12

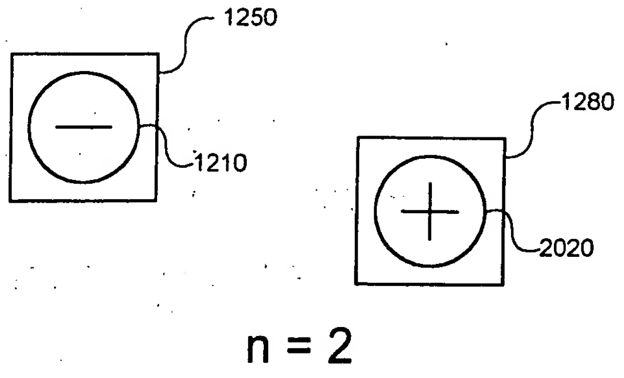


Figure 13a

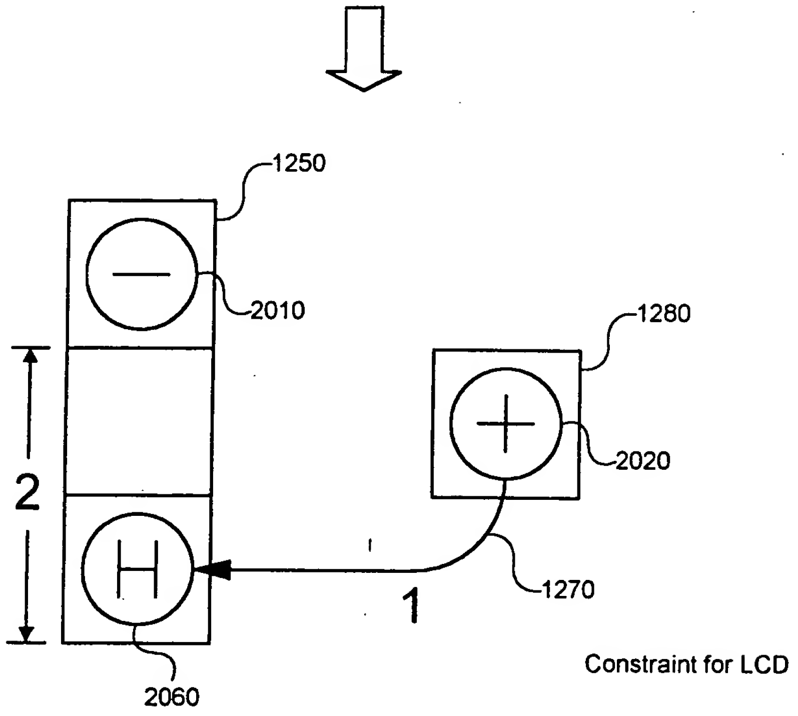


Figure 13b

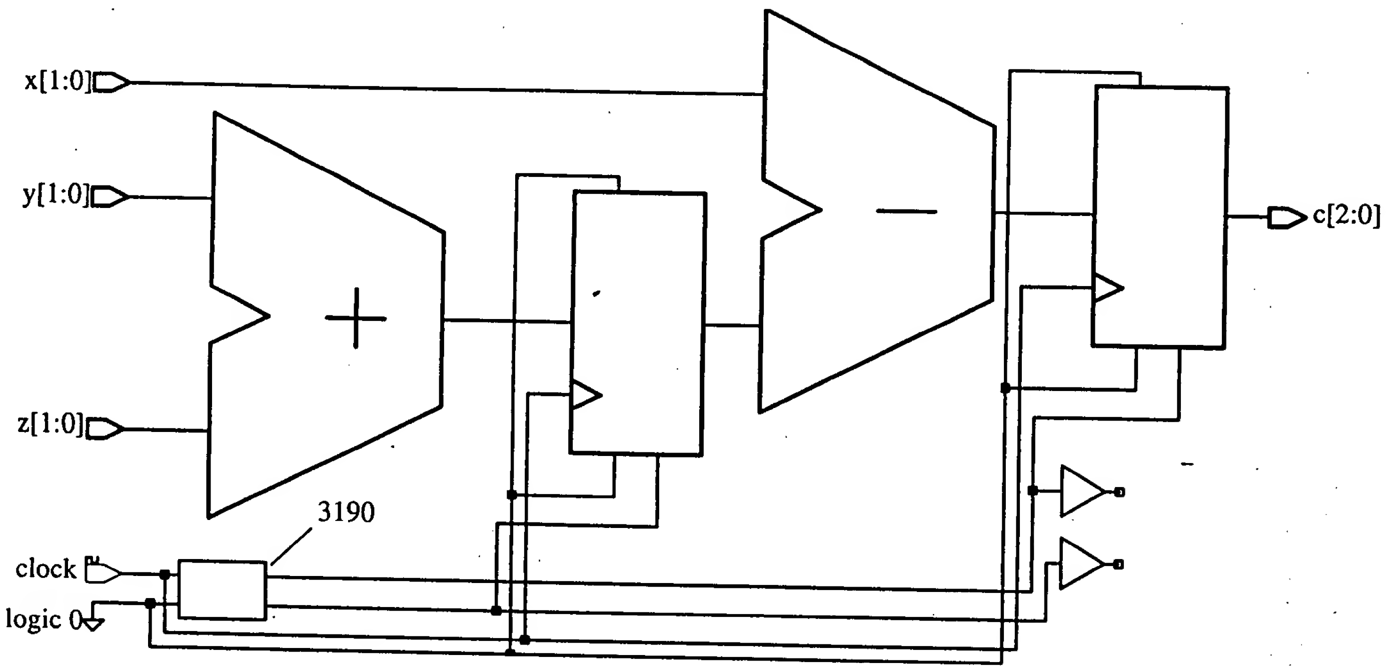
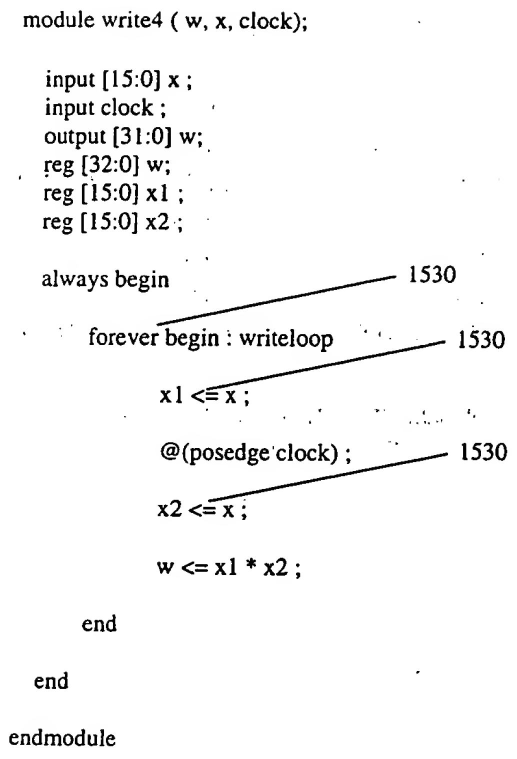


Figure 14



```
module write4 ( w, x, clock);

  input [15:0] x ;
  input clock ;
  output [31:0] w;
  reg [32:0] w;
  reg [15:0] x1 ;
  reg [15:0] x2 ;

  always begin

      forever begin : writeloop

          x1 <= x ;

          @(posedge clock) ;

          x2 <= x ;

          w <= x1 * x2 ;

      end

  end

endmodule
```

Figure 15

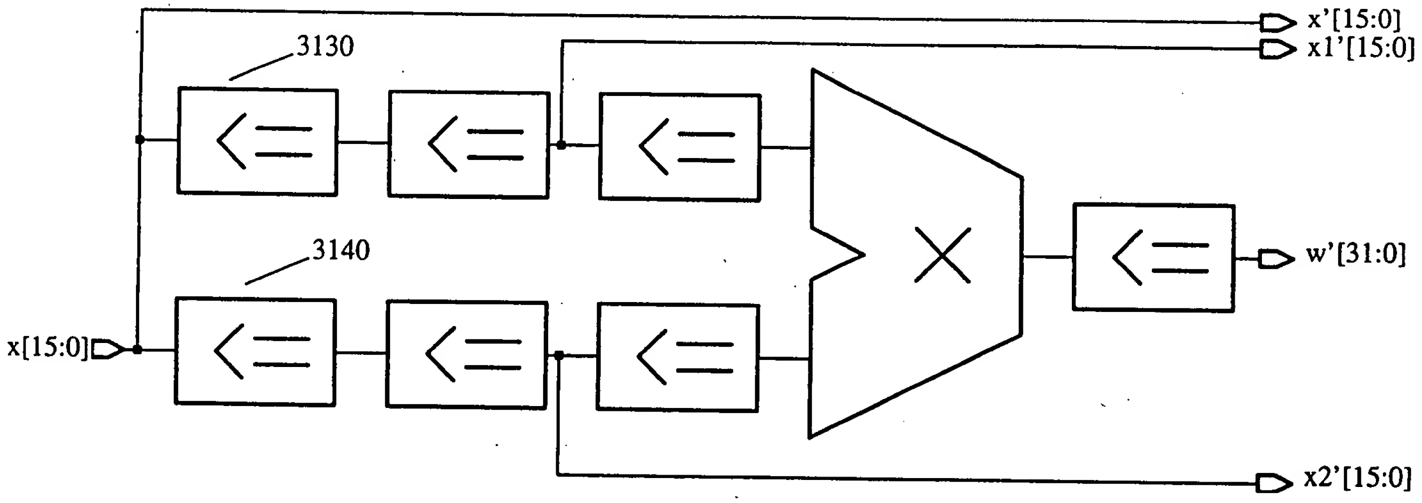


Figure 16

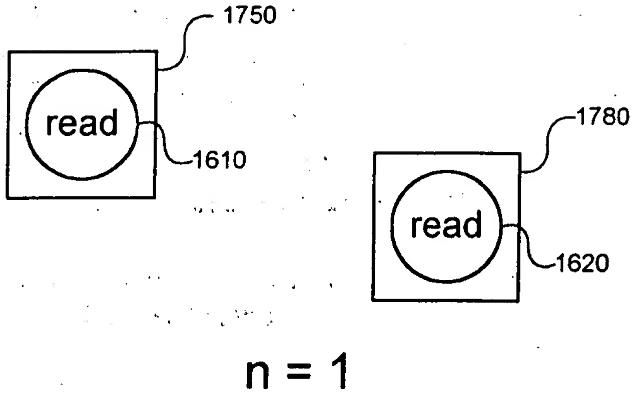


Figure 17a

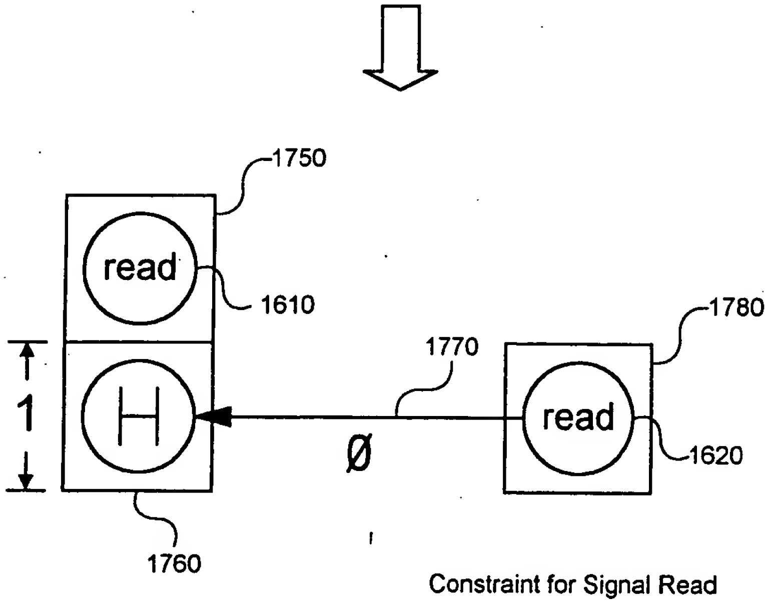


Figure 17b

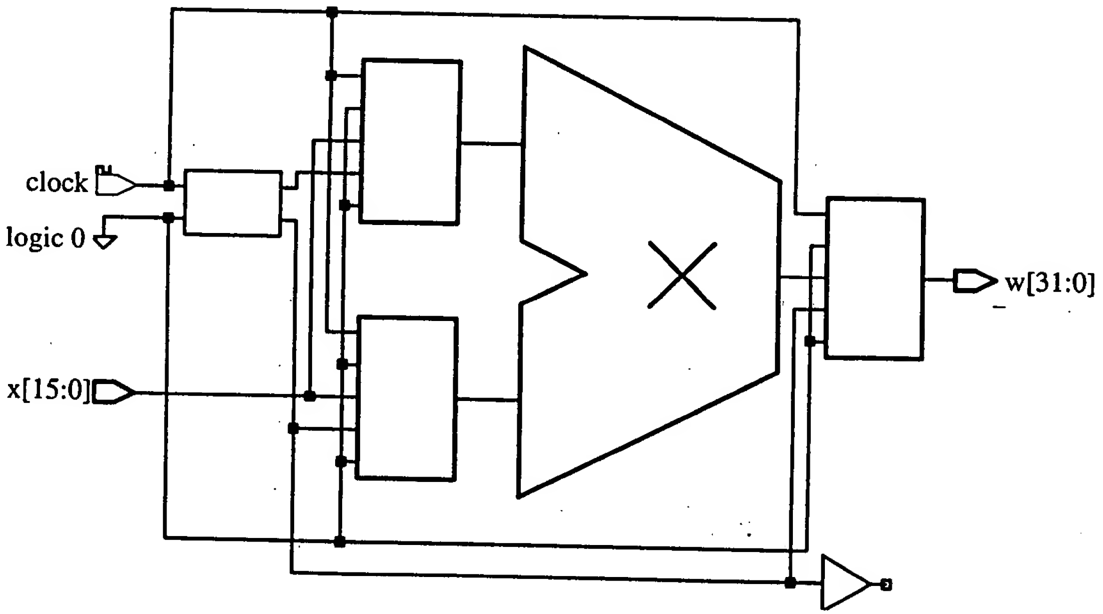


Figure 18

```
module after1 ( c, x, y, z, clock);

    input [1:0] x, y, z;
    input clock ;
    output [2:0] c;
    reg [2:0] c;
    reg [2:0] p;

    always begin

            @(posedge clock) ;

            forever begin
                    c <= #24 x - p ;

                    @(posedge clock) ;

                    p = y + z ;

                    @(posedge clock) ;
            end
    end
endmodule
```

Figure 19 (a)

```
entity after1 is
    port(
            c : out integer range 0 to 7;
            x, y, z : in integer range 0 to 3;
            clock : in bit
    );
end after1;

architecture behavioral of after1 is begin
    process
            variable p : integer range 0 to 7;
    begin
            wait until clock'event and clock = '1';

            loop

              c <= transport x - p after 24 ns;

              wait until clock'event and clock = '1';

              p := y + z;

              wait until clock'event and clock = '1';

            end loop;

    end process;
end behavioral;
```

Figure 19 (b)

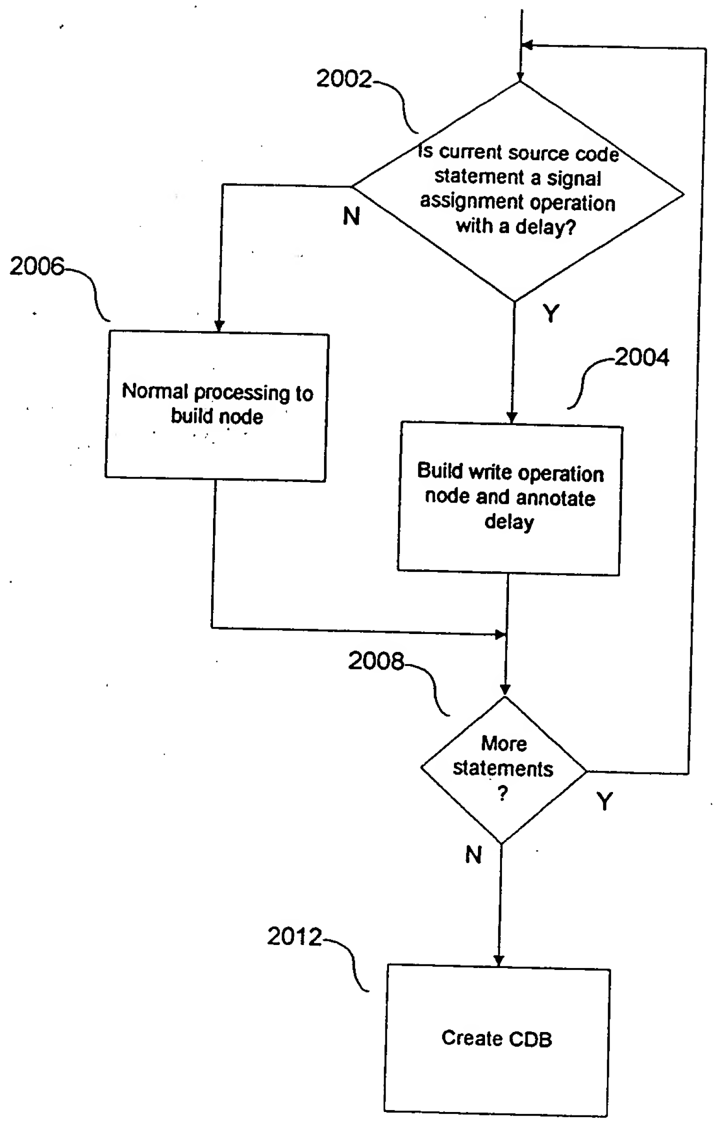

2002
Is current source code statement a signal assignment operation with a delay?
N
Y
2006
Normal processing to build node
2004
Build write operation node and annotate delay
2008
More statements ?
Y
N
2012
Create CDB

Fig. 20

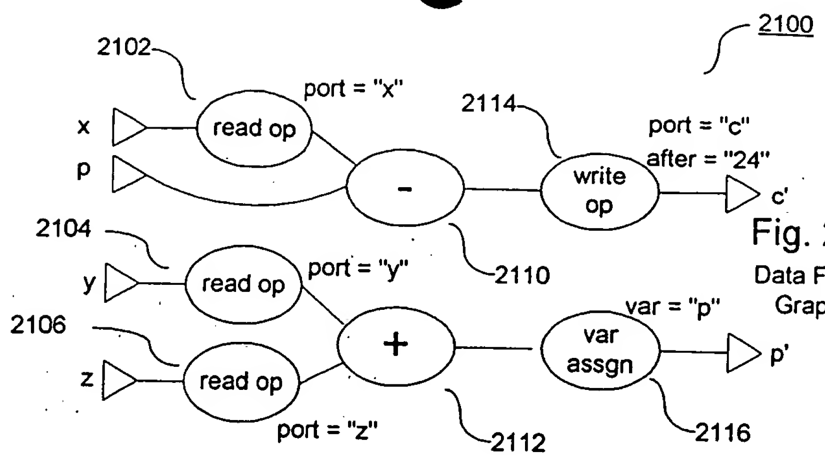


Fig. 21
Data Flow
Graph

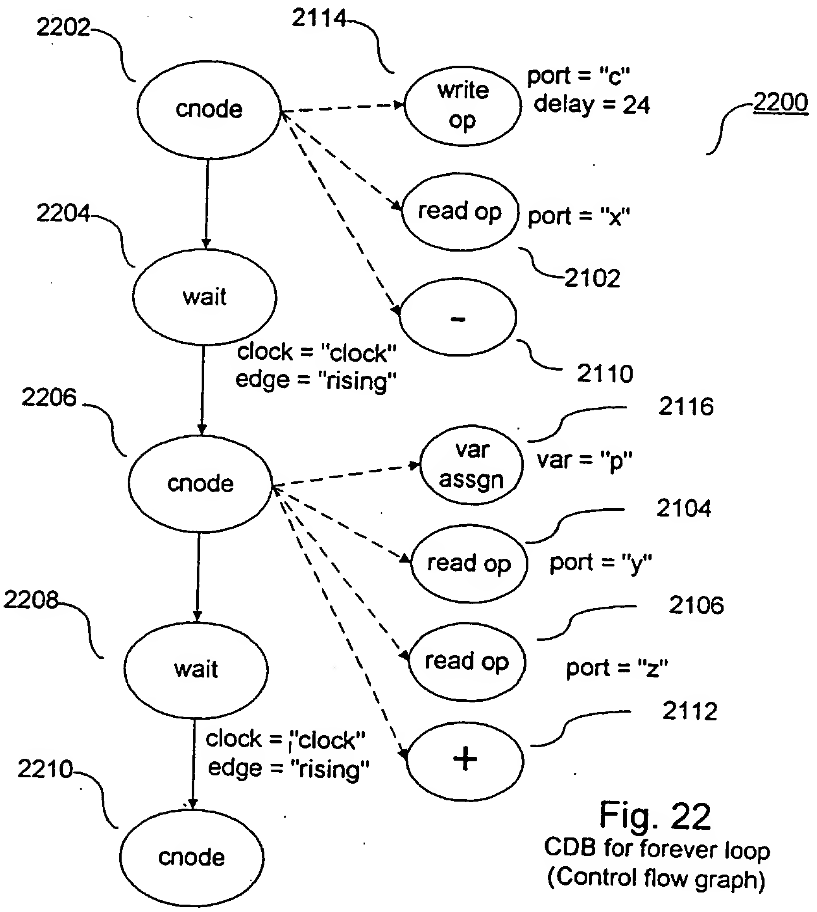
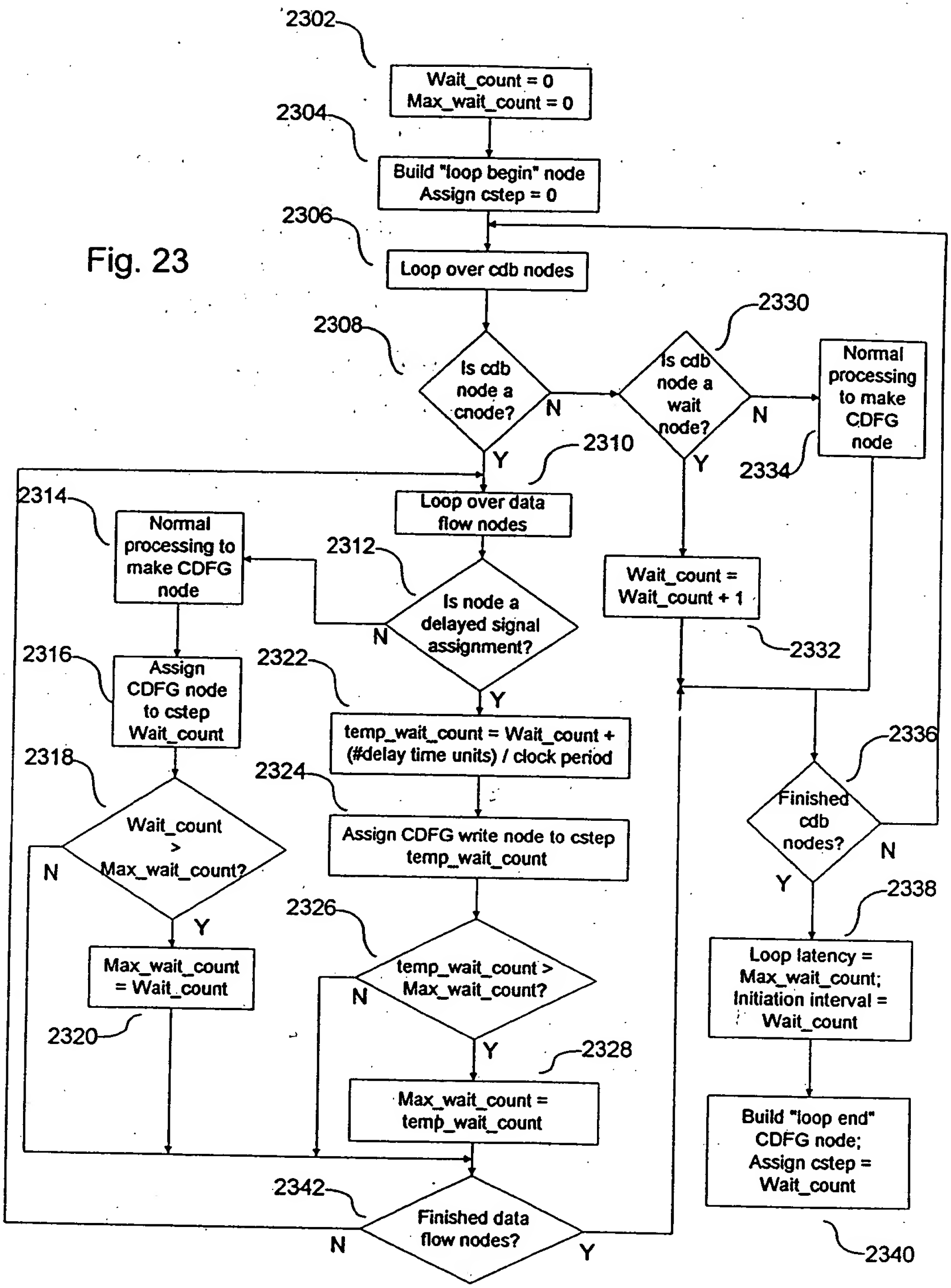


Fig. 22
CDB for forever loop
(Control flow graph)

Fig. 23


2302
Wait_count = 0
Max_wait_count = 0
2304
Build "loop begin" node
Assign cstep = 0
2306
Loop over cdb nodes
2308
Is cdb node a cnode?
N
Y
2330
Is cdb node a wait node?
N
Y
Normal processing to make CDFG node
2334
2310
Loop over data flow nodes
2314
Normal processing to make CDFG node
2312
Is node a delayed signal assignment?
N
Y
Wait_count = Wait_count + 1
2332
2316
Assign CDFG node to cstep Wait_count
2322
temp_wait_count = Wait_count + (#delay time units) / clock period
2318
Wait_count > Max_wait_count?
N
Y
2324
Assign CDFG write node to cstep temp_wait_count
2336
Finished cdb nodes?
N
Y
2338
Loop latency = Max_wait_count; Initiation interval = Wait_count
2326
temp_wait_count > Max_wait_count?
N
Y
Max_wait_count = Wait_count
2320
2328
Max_wait_count = temp_wait_count
Build "loop end" CDFG node; Assign cstep = Wait_count
2342
Finished data flow nodes?
N
Y
2340

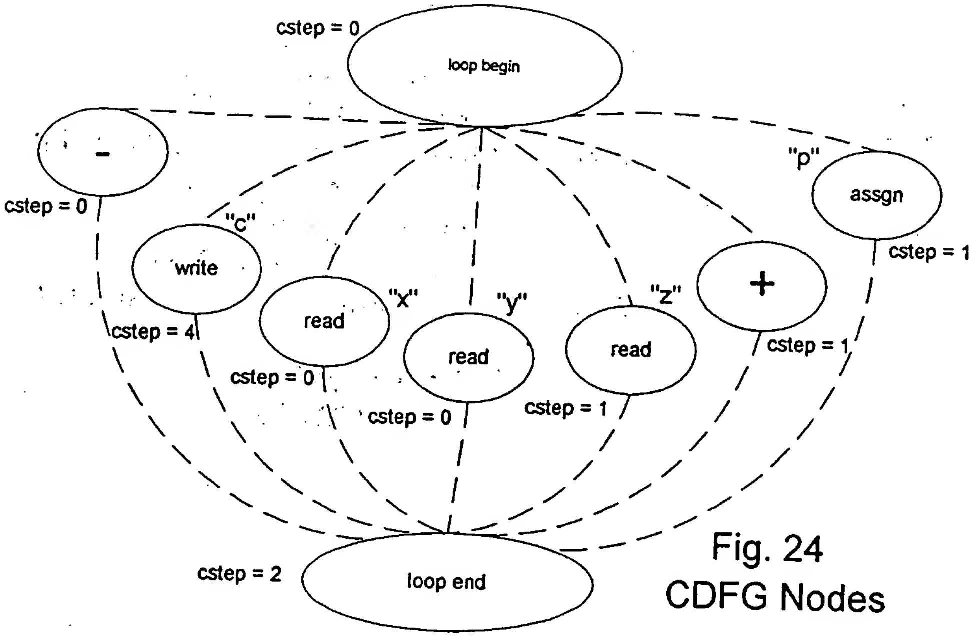


Fig. 24
CDFG Nodes

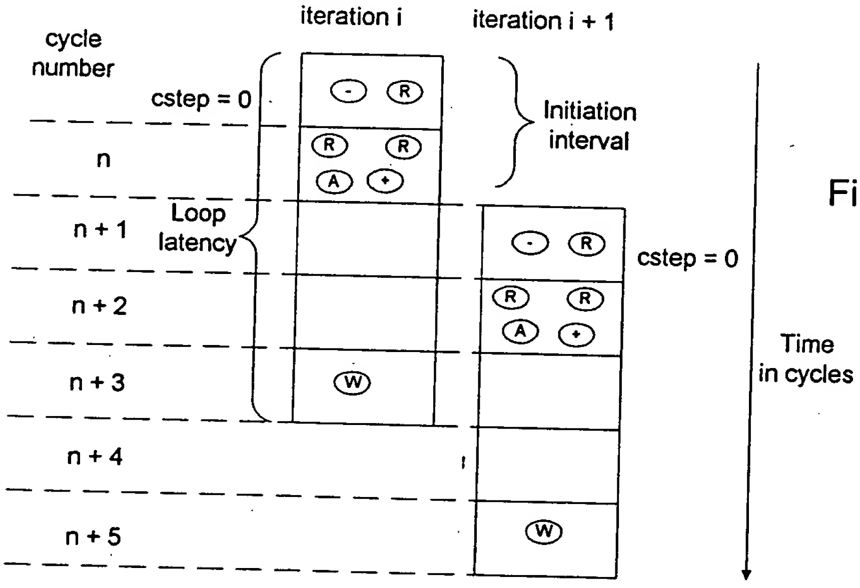


Fig. 26

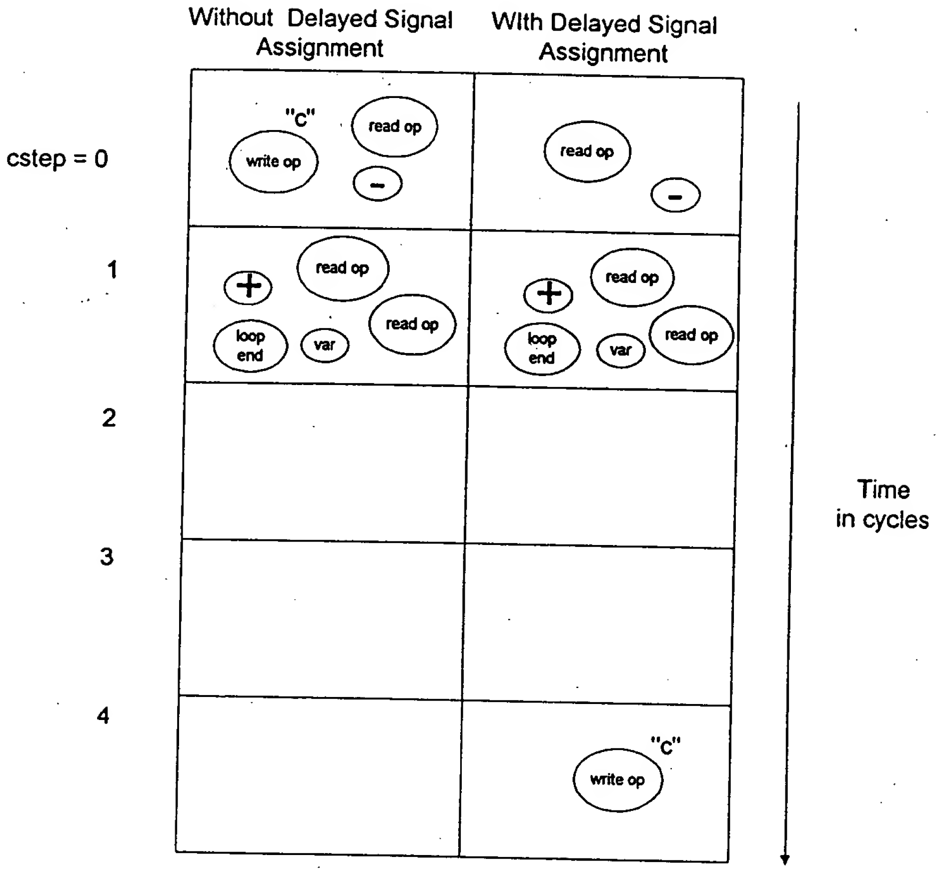

Without Delayed Signal Assignment
WIth Delayed Signal Assignment
cstep = 0
1
2
3
4
"c"
write op
read op
-
read op
-
+
read op
read op
loop end
var
+
read op
loop end
var
read op
"c"
write op
Time in cycles

Fig. 25

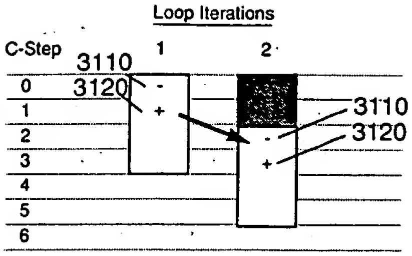


Figure 27

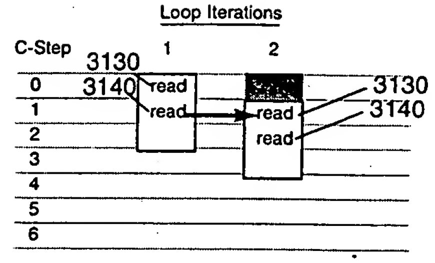

Loop Iterations
C-Step
1
2
0
1
2
3
4
5
6
3130
3140
read
read
read
read
3130
3140

Figure 28